Towards a Design Space for Blockchain-Based System Reengineering

Marco Comuzzi(✉), Erdenekhuu Unurjargal, and Chiehyeon Lim

Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea
{mcomuzzi,eeg1123,chlim}@unist.ac.kr

Abstract. We discuss our ongoing effort in designing a methodology for blockchain-based system reengineering. In particular, we focus in this paper on defining the design space, i.e., the set of options available to designers when applying blockchain to reengineer an existing system. In doing so, we use a practice-driven approach, in which this design space is constructed bottom-up from analysis of existing blockchain use cases and hands-on experience in real world design case studies. Two case studies are presented: using blockchain to reengineer the meat trade supply chain in Mongolia and blockchain-based management of ERP post-implementation modifications.

Keywords: Blockchain · System design · Reengineering

1 Introduction

A blockchain is a shared ledger through which different parties can verify and store any kind of records and transactions in blocks. A block has the hash information of the previous block and the blocks are connected with the hash information and form a chain. Owing to cryptographic techniques, a blockchain is immutable and tamper-proof by design and, as such, it creates trust among involved parties without the need for intermediaries to verify the validity of transactions. Although blockchain is recognised as a potentially revolutionary technology for all industries, deployment of blockchains outside finance and, specifically, cryptocurrencies and related applications, is still largely experimental. For instance, all papers in a special issue on blockchain and information systems research recently published in BISE[1] considered only applications of blockchain in risk management and finance.

In the literature, we recognise some efforts to look at blockchain as a design tool for reengineering existing systems. Benefits of applying blockchain in a system span from removing intermediaries and lowering transactional costs to

This work was supported by UNIST research fund, project number 1.180055.

[1] http://www.bise-journal.com/?p=1243.

increasing agility and security, and creating trust among partners. Kshetri [5] has recently analysed the potential impact of blockchain on supply chain reengineering. Their analysis, however, is limited to benefits and drawbacks and only marginally focuses on blockchain as a design element of a supply chain. In the field of software architecture, Xu et al. [9] provide a thorough taxonomy of blockchain-based systems for architecture design. The analysis, however, is situated at a technical, developer-oriented level, focusing on protocol design and computational efficiency. Nevertheless, as shown later, many of the considerations made by Xu et al. have also inspired our work.

Based on our analysis instantiated above, we argue that more research is needed about blockchain as a tool for system reengineering. In this paper we aim at giving an overview of our ongoing research in this direction. In particular, we describe our efforts in defining a methodology for blockchain-based system reengineering. Virtually any type of system where information is exchanged by multiple parties can be reengineered using blockchain. The wide applicability of blockchain as a reengineering paradigm makes it impossible to develop a top-down approach that could fit any possible scenario. Conversely, in this work we take a bottom-up, practice-driven approach, in which the design space of blockchain-based system reengineering is built by considering the features required by different real design cases analysed by the authors (see [7] for a similar process followed by the authors in the context of defining big data to advance service). At this nascent stage of blockchain literature, this empirical approach would be useful to derive theoretical implications. We concur with the Action Research philosophy [2] that close observation and understanding is possible only through action. Nothing helps understand blockchain-based system reengineering better than the reengineering of one made by the researchers.

In this paper, we present a first iteration in developing a blockchain-based system reengineering design space. This is based on a preliminary analysis of several blockchain use cases in different industries, and, most importantly, on two case studies that we are currently developing. These two case studies are briefly presented in the next section. Section 3 introduces the design space. Section 4 discusses ongoing and future work while drawing the conclusions.

2 Case Studies

Case 1 refers to reengineering the livestock supply chain in the Mongolian meat market; case 2 refers to using blockchain for managing post-implementation changes in ERP implementations. Before initiating the two case studies, we analysed existing use cases of blockchain (e.g., in finance [3], energy [4], consumer electronics [6], and manufacturing [1]) to initially identify some dimensions of system design that have been tested and extended through our own design case studies.

Case 1: Blockchain-Based Livestock Supply Chain in Mongolia
The total number of livestock (sheep, goat, cow, horse and camel) in Mongolia is about 60 million units, that is, about 20 times the country's population.

Mongolian livestock is mostly owned by herders, who roam seasonally in the vast Mongolian countryside. Most of the meat consumption, however, occurs in the country's capital Ulanbataar, where about 40% of the country's population lives. The trade between herders and the capital's meat sellers, e.g., supermarkets and local markets, is mediated by so-called *dealers*. Dealers are middlemen, usually family-based, with enough manpower to either transport large livestock quantities to the capital or slaughter livestock in the countryside and transport the carcasses to the capital.

This current meat trade supply chain has several problems. The system is not regulated and dealers have a staggering bargaining power, since they are the only ones that can match supply and demand. This is a problem particularly for herders, who cannot negotiate prices that would allow them to have decent standards of living. Meat sellers, on the other hand, have no visibility on the supply chain and rely completely on information from dealers, which has major issues for tracking delivery dates and quantities and guaranteeing the source of the meat. While the role of dealers cannot be physically eliminated, this is a typical scenario in which blockchain-based reengineering can clearly reduce the power of dealers and dramatically increase the level of data transparency and trust among actors in the supply chain.

The solution that we envision in this case considers multiple blockchains, one for each identified supply chain, i.e., a set of herders, middlemen and meat sellers repeatedly doing business together. The content of blocks is fairly straightforward, since only basic information about traded livestock and provenance has to be recorded. Simple smart contracts to adjust traded stock levels can also be included. Blockchains can be also used to track payments, possibly involving external actors such as banks or escrow services. In order to maintain trust among partners, a new block can be added to blockchains only if all parties involved in the supply chain agree to it. A further development of applying blockchain in this scenario concerns the creation of one global higher level blockchain that the national government can use to study and apply regulations to this key market in the Mongolian economy. This global blockchain may involve meat companies and authorities and scale up the scenario. It can then be used to enforce regulation, such as stricter controls and reporting policies for herdsmen and dealers exceeding certain levels of gross trade, or verification of meat seller provenance information based on data stored in the blockchain.

Case 2: Blockchain to Manage ERP Post Implementation Changes
Most organisations worldwide have already gone through one or more ERP implementations and found themselves in the so-called post-implementation phase. In this phase, the ERP systems may require changes beyond traditional corrective maintenance to address emerging business requirements, e.g., new government regulations or different warehouse management policies. These changes bear an impact on the static design structure and runtime of an ERP system. A change of a business object, for instance, may not be compatible with functions using it, making them practically unusable. In turn, these functions may be used in business processes. The instances currently running of these processes may

run into trouble if they have not passed the execution point at which these unusable new functions are required. Managing ERP post-implementation changes helps to avoid chaotic evolution of the system, therefore increasing ERP system and data quality.

In previous work, the authors have already proposed and evaluated a tool-supported methodology to manage ERP post-implementation changes effectively [8]. Following the principle of engineering change management, a change to an ERP system is first proposed. Then, before being approved, it is evaluated in terms of its design- and run-time impact on the existing system. Based on its impact, a change can be either approved or refused. Once approved, a change is then executed. Execution involves the actual implementation of changes and the transition of running process instances affected by the change to a safe termination.

The methodology described above can clearly benefit from a blockchain-based reengineering. In particular, for any given ERP installation, we envision a single blockchain keeping track of all post-implementation changes occurred since the go-live of the system. A block should contain all information related with a change, such as the business requirement addressed, the persons involved in the change (approval, implementation etc.), the impact of the change, and details about what has been changed and how, e.g., whether the ERP code base has been modified or simply some ERP configuration parameters have been changed. A block should also include smart contracts to enforce the safe termination of running process instances affected by a change. Finally, the consensus rules in place to accept a new block in the blockchain should embed the evaluation of the impact of a proposed change. In particular, we envision a combination of automated and voting-based consensus. The impact of a change can be evaluated automatically and a change may be automatically refused if it has a too large impact on the existing system. At the same time, however, a voting system can be implemented to keep into account the human aspect in this decision making process: a proposed change, for instance, although of limited impact on the system, may be rejected by managers because not aligned to other organisational policies. Alternatively, a change may be rejected if a sufficient share of the users involved in the processes affected by the change do not agree with it, for instance because it would be modify current practice too radically.

Compared to existing methodologies, blockchain-based implementation brings a set of additional benefits to this scenario. The history of post-implementation changes of an ERP system can be documented fully in a traceable and immutable way. Most importantly, the blockchain would allow to track not only *what* has been done, but also *how* a change has been implemented, e.g., both the process followed to propose and approve a change and the smart contracts that have been enforced after the change implementation would also be stored safely in the blockchain.

3 Design Space

Based on the case studies introduced above, we organise the blockchain-based system reengineering design space into a set of design dimensions, which belong to two areas, i.e., *Structure* and *Interaction*. The former groups design concerns related to the static content of blockchain(s) in a system, whereas the latter identifies concerns related with the behaviour of actors in a system when using the blockchain.

In the *Structure* area, we identify the following 4 design dimensions:

Cardinality. Concerns the numbers of blockchain to be implemented in a given domain. In case study 1, one blockchain for each *trade network*, i.e., group of trading herders, dealers, meat sellers, can be created; a global blockchain can be created with aggregated information to support regulation enforcement at government level. In case study 2, one particular blockchain is created for each ERP installation to track all changes occurred to it. Note that one organisation may run multiple ERP installations from different vendors at once.

Type. Concerns whether public/permissioned or private blockchains are considered. Private and permissioned blockchains can increase security and resiliency by leveraging public computational resources to assess block validity. Case study 1 can benefit from using public/permissioned blockchains because it concerns a more open scenario, in which dealers and meat sellers can dynamically join, whereas case study 2 represents a typical private case, in which information should only be shared within a specific organisational domain.

Content of Blocks. Concerns the information recorded in blocks; In case 1, records are simply about livestocks and trades, with elementary smart contracts adjusting trade balances and stock levels; in case 2, the content is more complex and smart contracts should be deployed to ensure safe termination of running process instances affected by process change.

Block Determination. Concerns the way in blocks are determined. Case study 1 may adopt a temporal approach, similar to the one adopted by cryptocurrencies, in which blocks are written at a given frequency to include all transactions occurred in a given time interval; changes in case study 2 may occur at a much lower frequency than transactions in case study 1, so case study 2 may adopt an event-based paradigm, in which a new block is written for each ERP change.

In the *Interaction* area, we identify the following 4 design dimensions:

Validation. Concerns the (possibly automated) rules to validate the content of a block. In case 1, these are simple conditions on account balances and stock levels . In case 2, more complex rules to evaluate the validity of a proposed change and its impact on the existing ERP system should be considered.

Consensus. Concerns the way in which parties reach consensus before including a new valid block in a blockchain. In case 1, all parties should agree to new block creation; in case 2, a voting system giving more importance to the opinion of users/managers with more experience should be in place.

Incentives. Concerns the method to motivate and promote the participation and interactions of network participants. In case 1, a token may be used for monetary incentives to the participants; in case 2, a token may not be appropriate because the blockchain should be fully private.

Permissions. Concerns the scheme regulating the ability of actors to propose and access content of a blockchain. In case 1, dealers, sellers and herders can propose new transactions, possibly manually, while government has a read-only access to data in blockchains. In case 2, a more complex scheme regulates access, i.e., only process owners and technical managers may propose new ERP changes, while users can only read information about past changes.

4 Conclusions

The work presented in this paper is currently being developed. On the one hand, we are defining more case studies to extend/refine the design space definition. In particular, we are considering blockchain-based reenegineering of several initiatives in our university campus, such as the reengineering using blockchain of research notebooks in natural sciences and a campus-wide cryptocurrency to secure and facilitate dorm fees payment and on-campus student subsistence. On the other hand, we are developing the methodology beyond the design space definition, to support system design on top of the defined design method. We are currently developing systematic guidelines to define alternative design configurations and an optimisation method to choose the optimal configuration based on notion of utility (e.g. selecting the combination of design options that maximizes the utility function of one specific participant or that balances the utility functions of different participants).

References

1. Abeyratne, S.A., Monfared, R.P.: A Blockchain Application in Energy. Loughborough University (2017)
2. Avison, D.E., Lau, F., Myers, M.D., Nielsen, P.A.: Action research. Commun. ACM **42**, 87–94 (1999)
3. Guo, Y., Liang, C.: Blockchain application and outlook in the banking industry. Financ. Innov. **2**, 24 (2016)
4. Hukkinen, T., Mattila, J., Ilomäki, J., Seppala, T.: A Blockchain Application in Energy. The Research Institute of the Finnish Economy (2017)
5. Kshetri, N.: Blockchain's roles in meeting key supply chain management objectives. Int. J. Inf. Manag. **39**, 80–89 (2018)
6. Lee, J.H., Pilkington, M.: How the blockchain revolution will reshape the consumer electronics industry. IEEE Consum. Electron. Mag. **6**, 19–23 (2017)
7. Lim, C., Kim, K., Kim, M.J., Heo, J.Y., Kim, K.J., Maglio, P.P.: From data to value: a nine-factor framework for data-based value creation in information-intensive services. Int. J. Inf. Manag. **39**, 121–135 (2018)
8. Parhizkar, M., Comuzzi, M.: Impact analysis of ERP post-implementation modifications: design, tool support and evaluation. Comput. Ind. **84**, 25–38 (2017)
9. Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., Pautasso, C., Rimba, P.: A taxonomy of blockchain-based systems for architecture design. In: 2017 IEEE International Conference on Software Architecture (ICSA), pp. 243–252. IEEE (2017)